FIG. 1

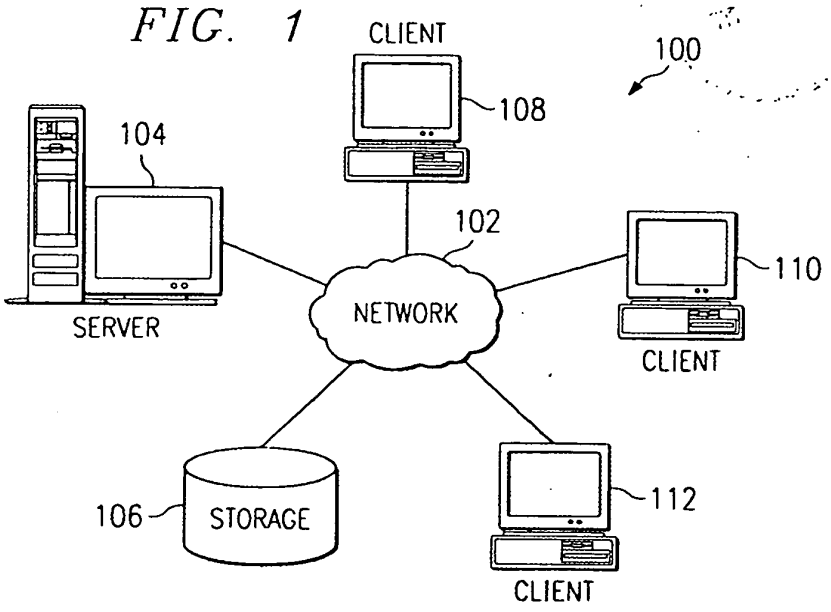
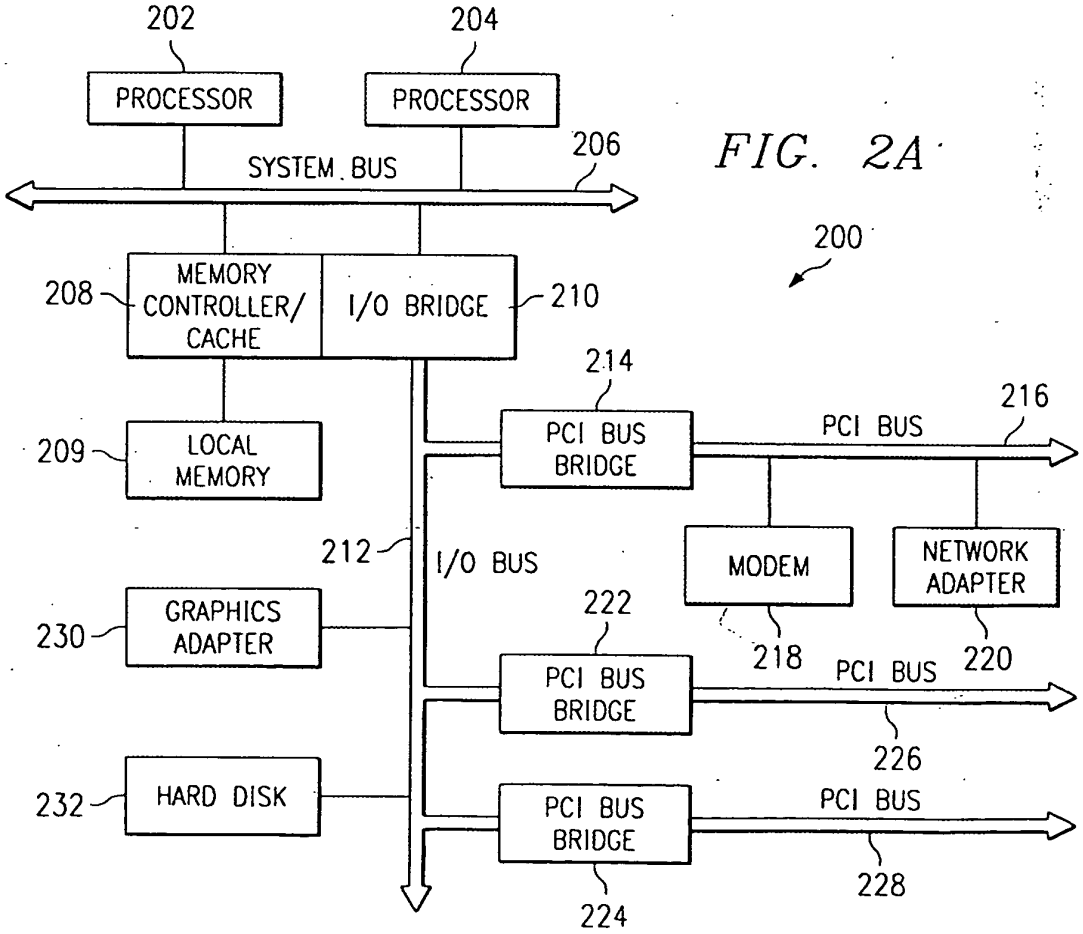

CLIENT
100
108
104
102
110
SERVER
NETWORK
CLIENT
106
STORAGE
112
CLIENT

FIG. 2A


202
204
PROCESSOR
PROCESSOR
206
SYSTEM BUS
200
208
MEMORY CONTROLLER/ CACHE
I/O BRIDGE
210
214
216
209
LOCAL MEMORY
PCI BUS BRIDGE
PCI BUS
212
I/O BUS
MODEM
NETWORK ADAPTER
222
230
GRAPHICS ADAPTER
218
PCI BUS
220
PCI BUS BRIDGE
226
232
HARD DISK
PCI BUS BRIDGE
PCI BUS
228
224

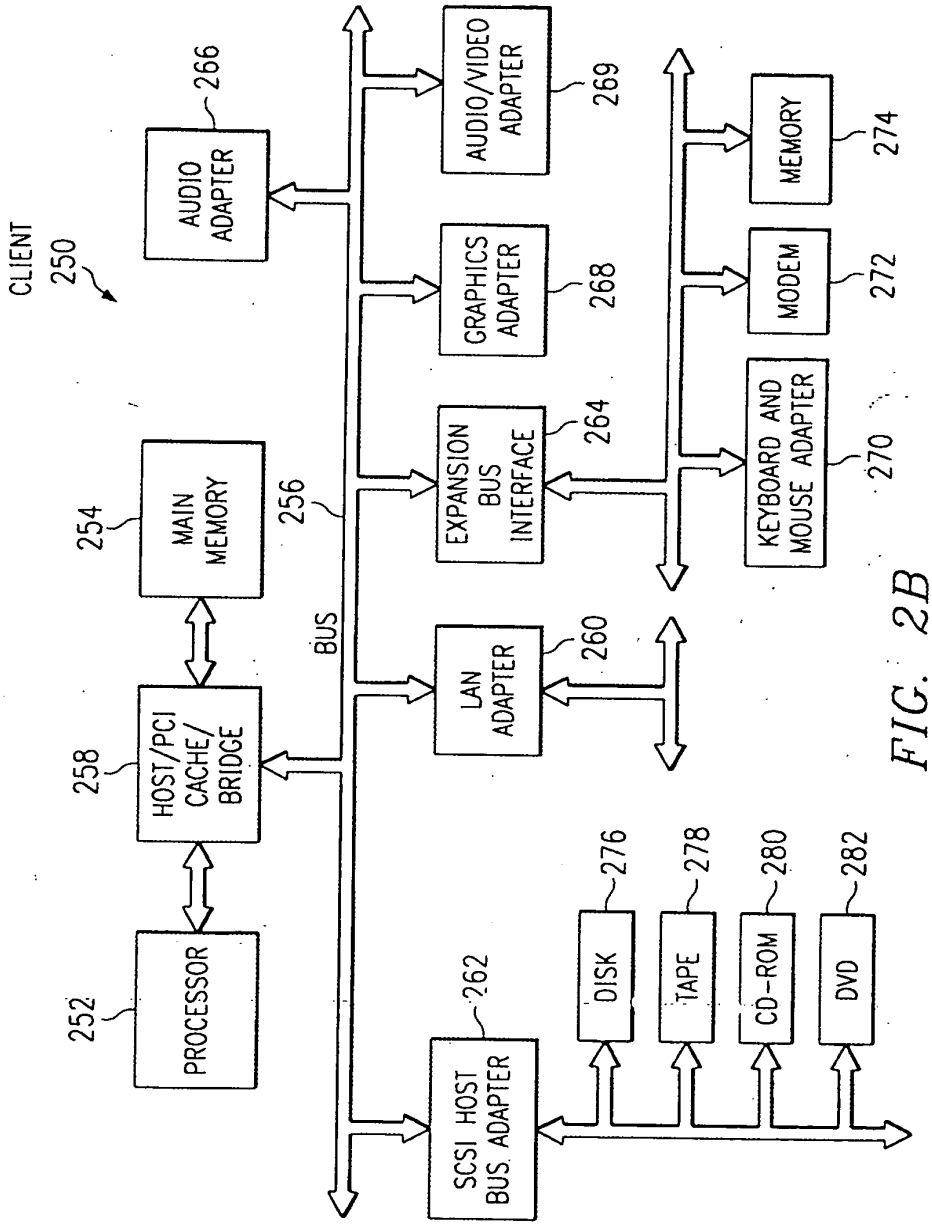

CLIENT
250
PROCESSOR
252
HOST/PCI CACHE/ BRIDGE
258
MAIN MEMORY
254
BUS
256
AUDIO ADAPTER
266
AUDIO/VIDEO ADAPTER
269
GRAPHICS ADAPTER
268
EXPANSION BUS INTERFACE
264
LAN ADAPTER
260
SCSI HOST BUS ADAPTER
262
DISK
276
TAPE
278
CD-ROM
280
DVD
282
MEMORY
274
MODEM
272
KEYBOARD AND MOUSE ADAPTER
270

FIG. 2B

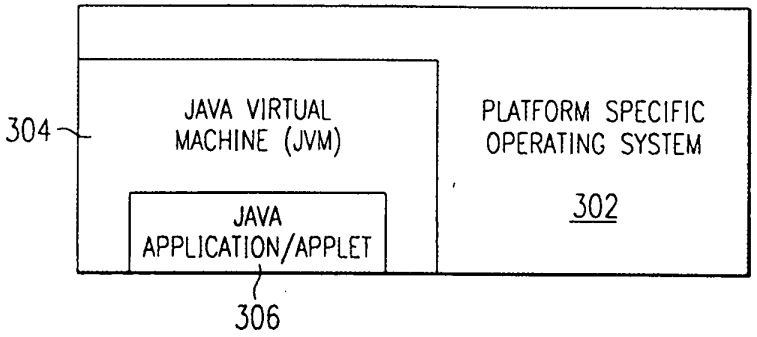

300
304
JAVA VIRTUAL MACHINE (JVM)
JAVA APPLICATION/APPLET
306
PLATFORM SPECIFIC OPERATING SYSTEM
302

FIG. 3A

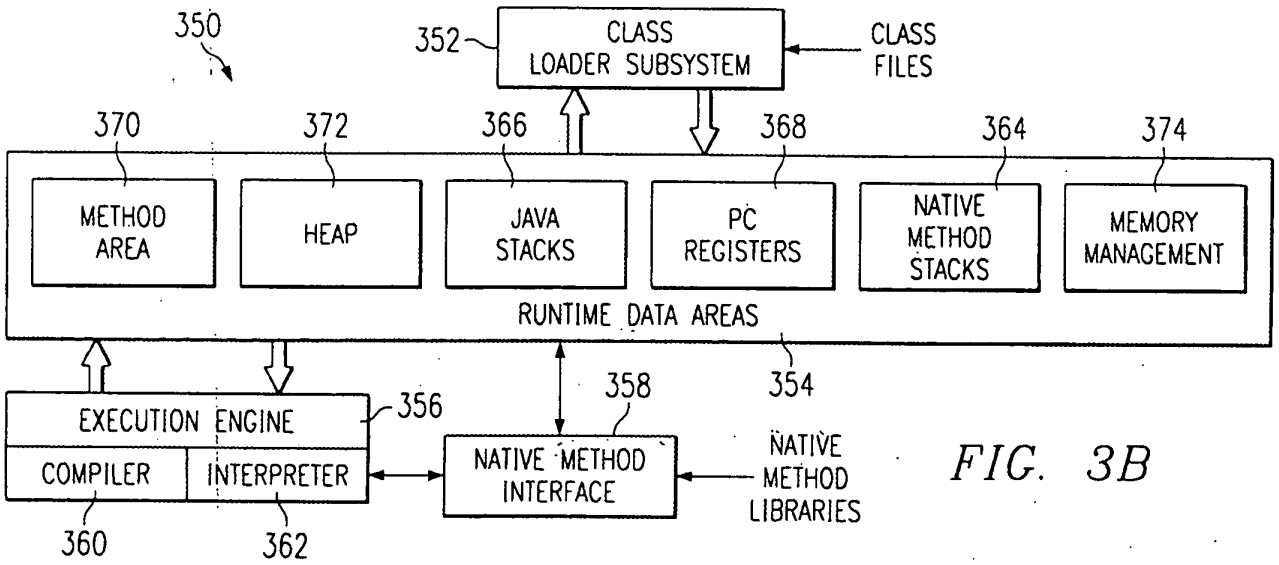

350
352
CLASS LOADER SUBSYSTEM
CLASS FILES
370
372
366
368
364
374
METHOD AREA
HEAP
JAVA STACKS
PC REGISTERS
NATIVE METHOD STACKS
MEMORY MANAGEMENT
RUNTIME DATA AREAS
354
358
356
EXECUTION ENGINE
COMPILER
INTERPRETER
NATIVE METHOD INTERFACE
NATIVE METHOD LIBRARIES
360
362

FIG. 3B

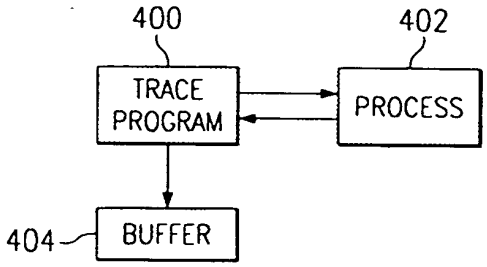


FIG. 4

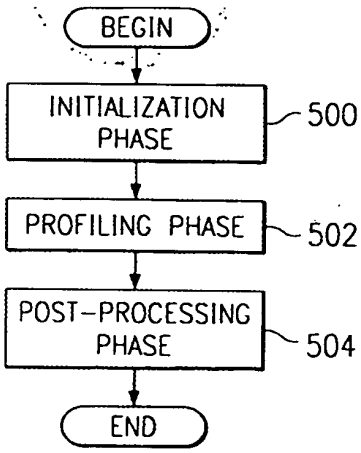


FIG. 5

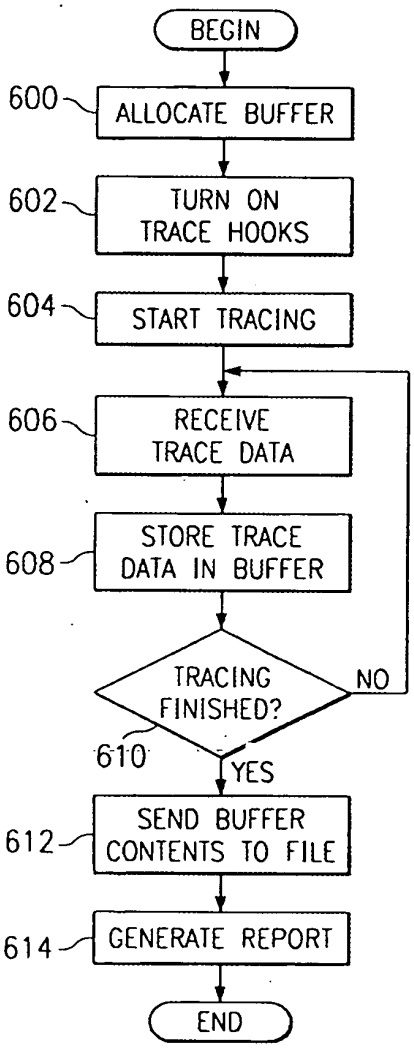


FIG. 6

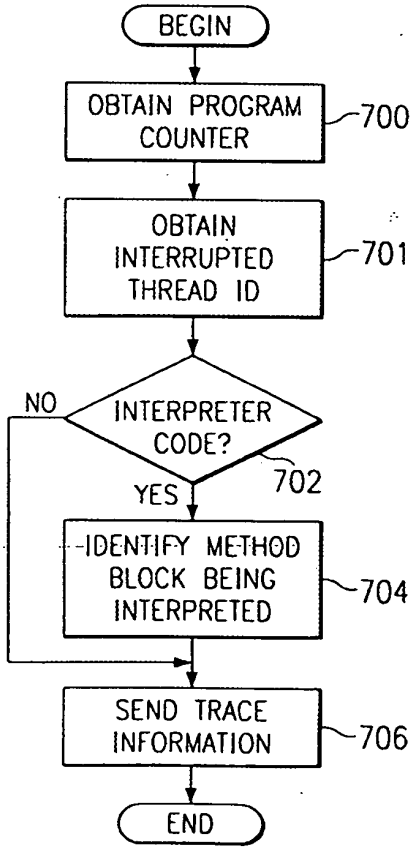


FIG. 7

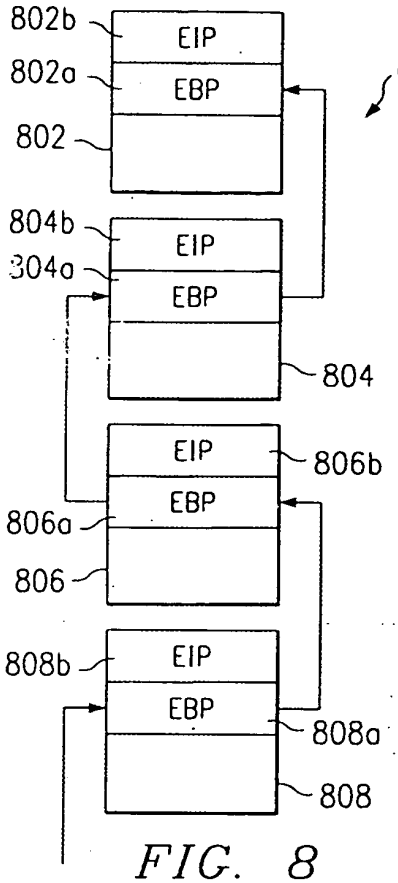


FIG. 8

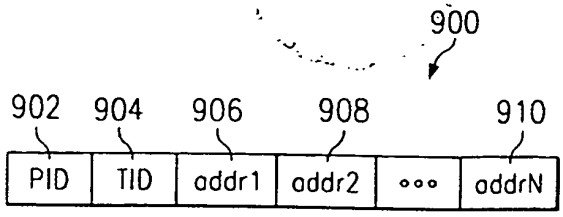


FIG. 9

SAMPLE	CALL STACK @ SAMPLE
1	C
2	CAB
3	CAB
4	CAB
5	C
6	CBA
7	CBABA
8	CBA
9	CBA
10	C

FIG. 10B

TIMESTAMP	EVENT	CALL STACK AFTER EVENT
0	ENTER C	
1	ENTER A	C
2	ENTER B	CA
3	EXIT FROM B	CAB
4	ENTER B	CA
5	ENTER B	CAB
6	EXIT FROM B	CABB
7	EXIT FROM B	CAB
8	EXIT FROM A	CA
9	ENTER B	C
10	ENTER A	CB
11	ENTER B	CBA
12	ENTER A	CBAB
13	EXIT FROM A	CBABA
14	EXIT FROM B	CBA
15	ENTER X	CBAX
16	EXIT FROM X	CBA
17	EXIT FROM A	CB
18	EXIT FROM B	C
19	EXIT FROM C	

FIG. 10A

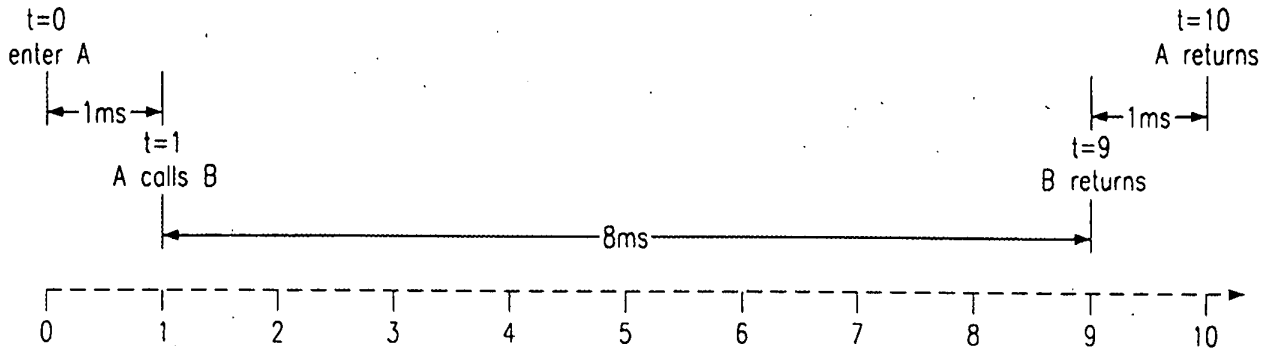


FIG. 10C

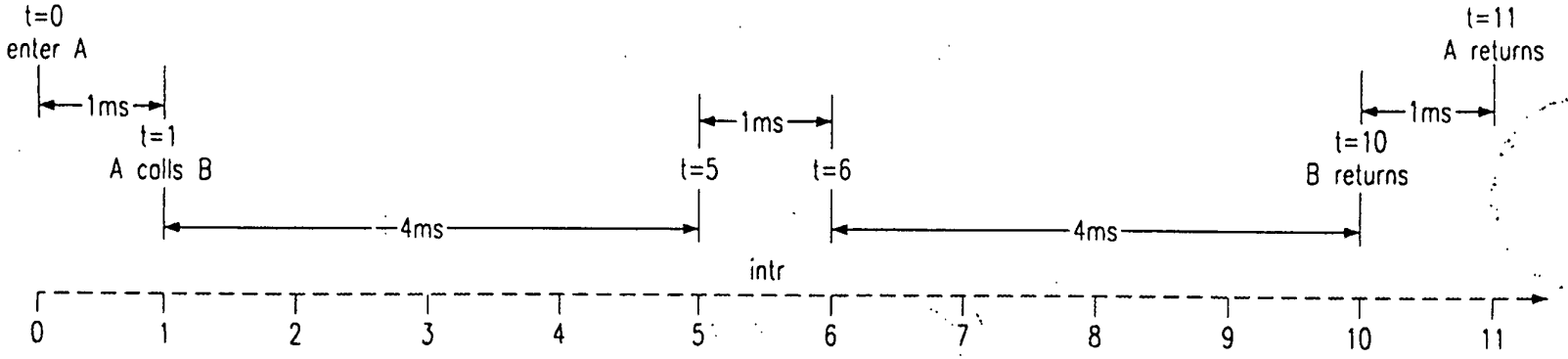


FIG. 10D

FIG. 11A

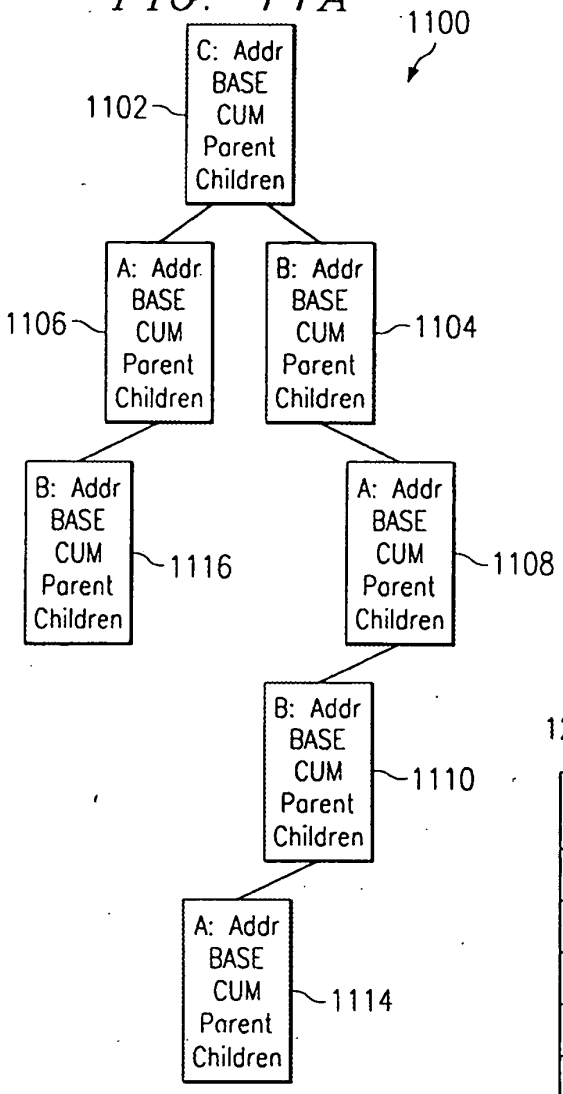


FIG. 11B

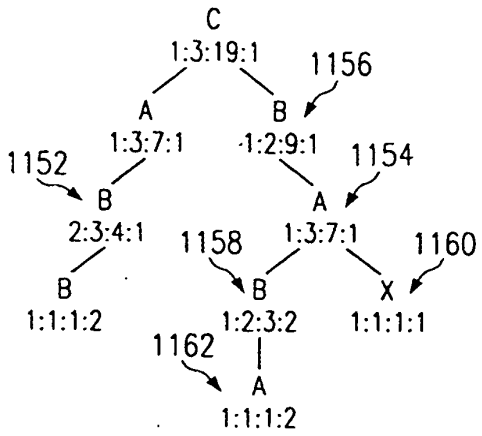


FIG. 12

1230 LEVEL	1232 RL	1234 CALLS	1236 BASE	1238 CUM	1240 INDENT
0	1	1	0	19	pt_pidtid
1	1	1	3	19	-C
2	1	1	3	7	--A
3	1	2	3	4	---B
4	2	1	1	1	----B
2	1	1	2	9	--B
3	1	1	3	7	---A
4	2	1	2	3	----B
5	2	1	1	1	-----A
4	1	1	1	1	----X

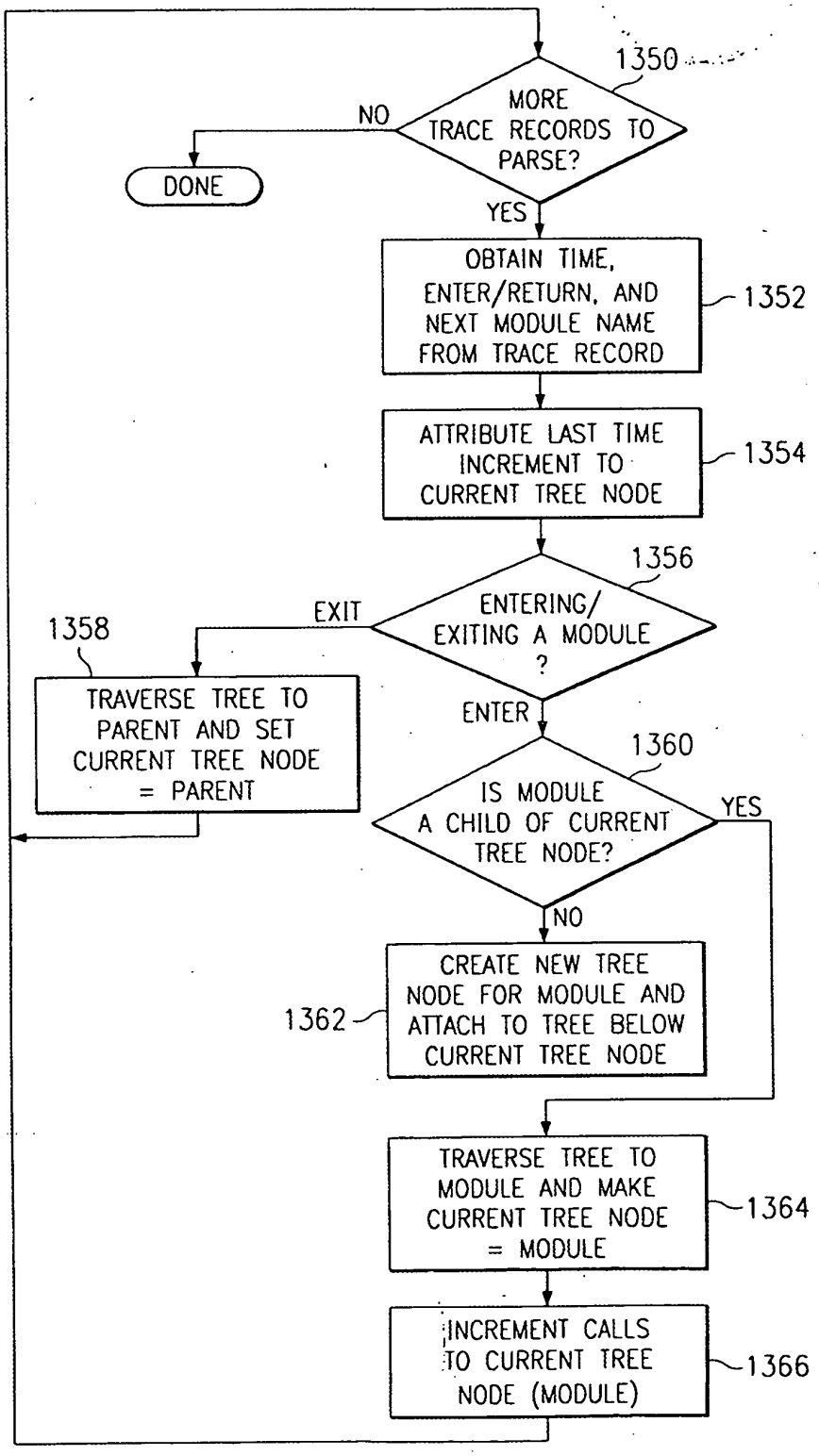

1350
MORE TRACE RECORDS TO PARSE?
NO
DONE
YES
OBTAIN TIME, ENTER/RETURN, AND NEXT MODULE NAME FROM TRACE RECORD
1352
ATTRIBUTE LAST TIME INCREMENT TO CURRENT TREE NODE
1354
1356
ENTERING/ EXITING A MODULE ?
EXIT
1358
TRAVERSE TREE TO PARENT AND SET CURRENT TREE NODE = PARENT
ENTER
1360
IS MODULE A CHILD OF CURRENT TREE NODE?
YES
NO
CREATE NEW TREE NODE FOR MODULE AND ATTACH TO TREE BELOW CURRENT TREE NODE
1362
TRAVERSE TREE TO MODULE AND MAKE CURRENT TREE NODE = MODULE
1364
INCREMENT CALLS TO CURRENT TREE NODE (MODULE)
1366

FIG. 13

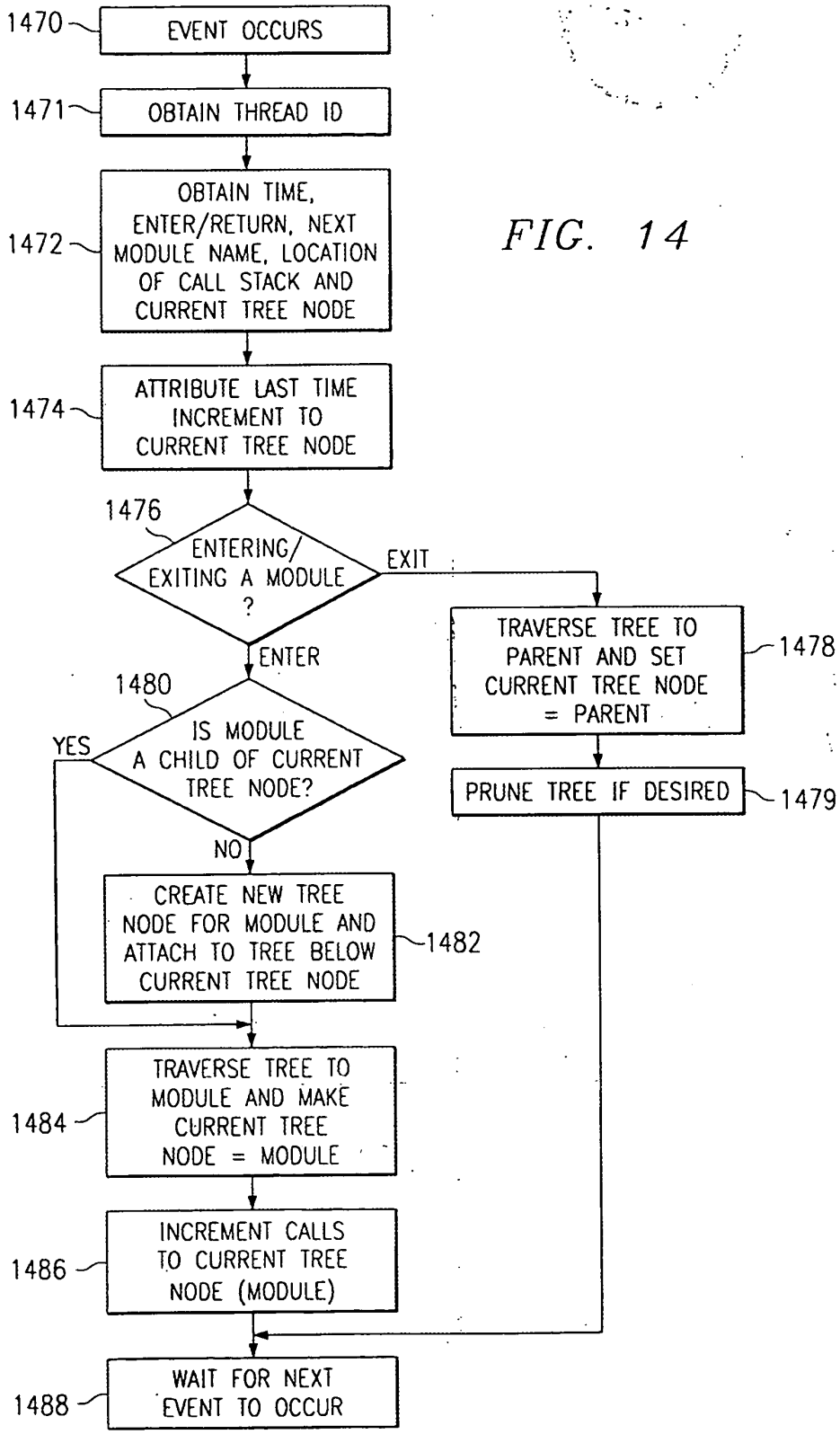

1470
EVENT OCCURS
1471
OBTAIN THREAD ID
1472
OBTAIN TIME, ENTER/RETURN, NEXT MODULE NAME, LOCATION OF CALL STACK AND CURRENT TREE NODE
1474
ATTRIBUTE LAST TIME INCREMENT TO CURRENT TREE NODE
1476
ENTERING/ EXITING A MODULE ?
EXIT
ENTER
1478
TRAVERSE TREE TO PARENT AND SET CURRENT TREE NODE = PARENT
1479
PRUNE TREE IF DESIRED
1480
IS MODULE A CHILD OF CURRENT TREE NODE?
YES
NO
1482
CREATE NEW TREE NODE FOR MODULE AND ATTACH TO TREE BELOW CURRENT TREE NODE
1484
TRAVERSE TREE TO MODULE AND MAKE CURRENT TREE NODE = MODULE
1486
INCREMENT CALLS TO CURRENT TREE NODE (MODULE)
1488
WAIT FOR NEXT EVENT TO OCCUR

FIG. 14

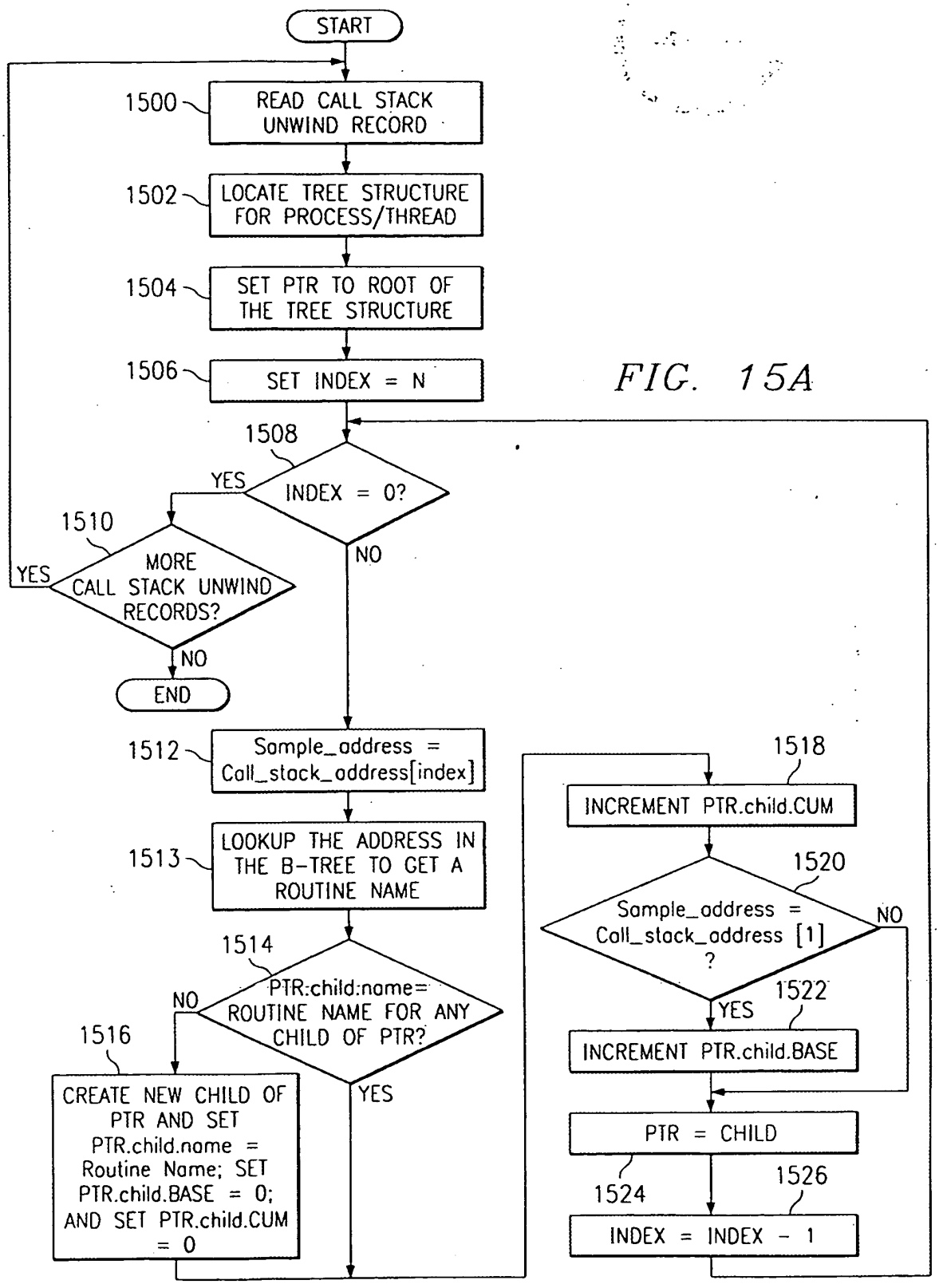

START
1500
READ CALL STACK UNWIND RECORD
1502
LOCATE TREE STRUCTURE FOR PROCESS/THREAD
1504
SET PTR TO ROOT OF THE TREE STRUCTURE
1506
SET INDEX = N
1508
INDEX = 0?
YES
NO
1510
MORE CALL STACK UNWIND RECORDS?
YES
NO
END
1512
Sample_address = Call_stack_address[index]
1513
LOOKUP THE ADDRESS IN THE B-TREE TO GET A ROUTINE NAME
1514
PTR:child:name= ROUTINE NAME FOR ANY CHILD OF PTR?
NO
YES
1516
CREATE NEW CHILD OF PTR AND SET PTR.child.name = Routine Name; SET PTR.child.BASE = 0; AND SET PTR.child.CUM = 0
1518
INCREMENT PTR.child.CUM
1520
Sample_address = Call_stack_address [1] ?
NO
YES
1522
INCREMENT PTR.child.BASE
PTR = CHILD
1524
1526
INDEX = INDEX - 1

FIG. 15A

FIG. 15B

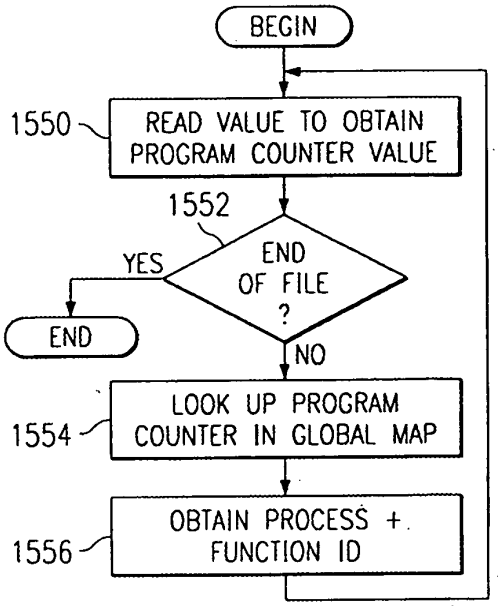
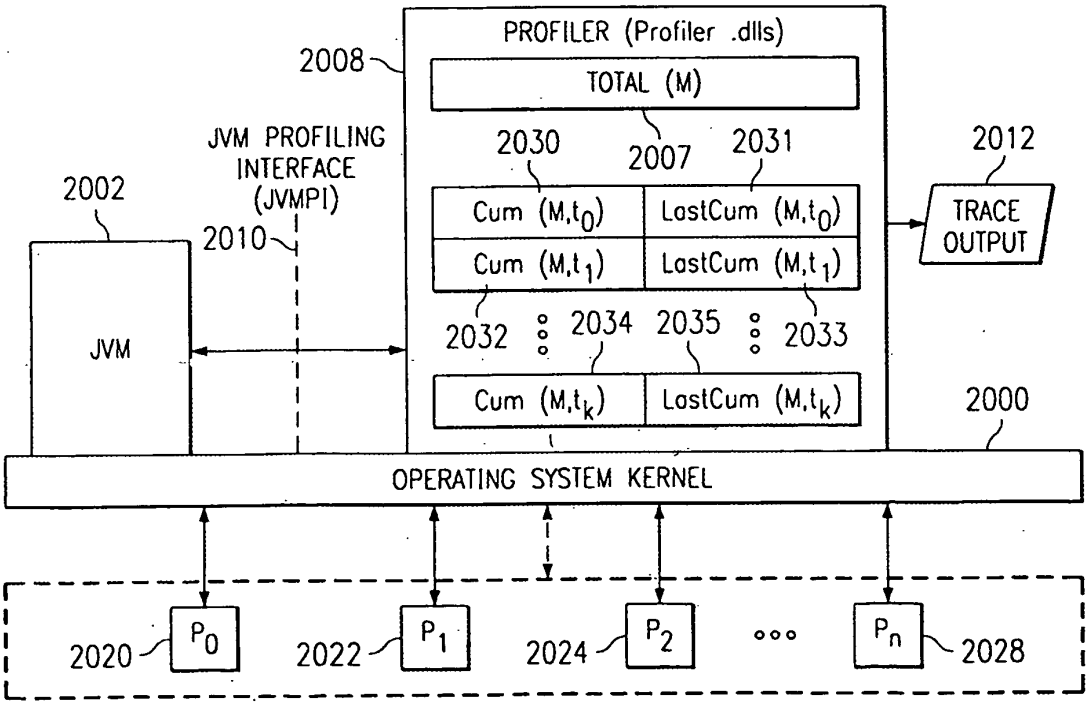


FIG. 16

1600

1602 CALLS	1604 BASE	1606 CUM	1608 NAME
1	0	19	pt_pit.tid
1	3	19	C
3	7	14	A
5	8	13	B
1	1	1	X

FIG. 20



1700

ArcFlow Output

Base – Time/Instructions directly in function
Cum – Time/Instructions directly & indirectly in function

ArcFlow Invarients:

1) Sum(Parent(Calls)) = Self(Calls)
2) Sum(Parent(Base)) = Self(Base)
3) Sum(Parent(Cum)) = Self(Cum)
4) Sum(Child(Cum)) = Self(Cum)–Self(Base)

Source	Calls	Base	Cum	Function
Self	1	0	19	[0] pt_pidtid
Child	1	3	19	C
Parent	1	3	19	pt_pidtid
Self	1	3	19	[1] C
Child	1	2	9	B
Child	1	3	7	A
Parent	1	3	7	C
Parent	1	3	7	B
rParent	1	1	1	B
Self	3	7	17 15	[2] A
Child	3	5	7	B
Child	1	1	1	X
Parent	2	3	4	A
rParent	1	2	3	A
Parent	1	2	9	C
Self	5	8	13 17	[3] B
Child	1	3	7	A
rChild	1	1	1	A
Child	1	1	1	B
Parent	1	1	1	A
Self	1	1	1	[4] X

FIG. 17

1800

Units : : Ticks
Total : : 342

LvL	RL	Calls	Base	Cum	Indent Name
1	1	1	0	342	- _Thread-21__(0xe0046618)
2	1	3	0	342	-- J:nulltestScore()I
3	1	2	0	272	--- J:nulltestMilliseconds(I)I
4	1	29450	0	271	---- J:nullexecute()I
5	1	271	0	271	----+ stack_0x40
6	1	271	0	271	----+- F:ExecuteJava ← 1802
7	1	271	0	271	----+-- F:jit_invokeCompiledEntryMethod
8	1	271	0	271	----+--- F:_jit_invokeentry
9	1	271	0	271	----+---- F:JITInvokeCompiledEntryMethod_md
10	1	271	0	271	----+----+ J:nullrun()V
11	2	271	0	271	----+----+- J:nulltestScore()I
12	2	271	0	271	----+----+-- J:nulltestMilliseconds(I)I
13	2	271	268	271	----+----+--- J:nullexecute()I
14	1	2	0	2	----+----+---- F:jperf_methodEntry
15	1	2	0	2	----+----+----+ F:SoftTracehook
16	1	2	2	2	----+----+----+- F:enable_interrupts
14	1	1	1	1	----+----+---- F:jperf_methodExit
4	1	1	0	1	---- stack_0x40
5	1	1	0	1	----+ F:ExecuteJava ← 1804
6	1	1	0	1	----+- F:jit_invokeCompiledEntryMethod
7	1	1	0	1	----+-- F:_jit_invokeentry
8	1	1	0	1	----+--- F:JITInvokeCompiledEntryMethod_md
9	1	1	0	1	----+---- J:nullrun()V
10	2	1	0	1	----+----+ J:nulltestScore()I
11	2	1	0	1	----+----+- J:nulltestMilliseconds(I)I
12	1	1	0	1	----+----+-- J:nullexecute()I
13	1	1	0	1	----+----+--- F:jperf_methodExit
14	1	1	0	1	----+----+---- F:SoftTracehook
15	1	1	1	1	----+----+----+ F:enable_interrupts
4	1	2	0	0	---- J:nullcleanUp()I

1806

1808

FIG. 18

FIG. 19

MAJOR CODE	MINOR CODE	DATA ITEM 1	DATA ITEM 2	DATA ITEM 3	DATA ITEM 4	DATA ITEM 5	DESCRIPTION
0x4	LEVEL + 0x1	DEPTH	N/A	N/A	N/A		BEGIN INTERRUPT AT LEVEL
0x4	LEVEL + 0x80000001	DEPTH	N/A	N/A	N/A		END INTERRUPT AT LEVEL
0x10	0xab	SYSTEM TID	JAVA TID	IS SYSTEM THREAD (BOOLEAN)	N/A		THREAD CREATED WITHOUT A NAME WHILE TRACE ACTIVE
0x10	0xac	SYSTEM TID	N/A	N/A	N/A		IDENTIFIES THE IDLE THREAD
0x10	0xad	SYSTEM TID	N/A	N/A	N/A		IDENTIFIES THE GARBAGE COLLECTION THREAD
0x10	0xae	SYSTEM TID	JAVA TID	THREAD NAME	N/A		THREAD CREATED WITH A NAME WHILE TRACE ACTIVE
0x30	0x10	OBJECT ID	METHOD BLOCK ADDRESS	N/A	N/A		METHOD INVOCATION (INTERPRETED)
0x30	0x10 + 0x80000000	OBJECT ID	METHOD BLOCK ADDRESS	N/A	N/A		METHOD EXIT (INTERPRETED)
0x40	0x7fffffff	NUMBER (n) OF STACK UNWINDS AT TIMER INTERRUPT	pc1-PROGRAM COUNTER OF INTERRUPTED ROUTINE	pc2-CALLER OF INTERRUPTED ROUTINE	.....	pcn-1 OF n-2nd CALLER OF INTERRUPTED ROUTINE	pcn OF n-1st CALLER OF INTERRUPTED ROUTINE
0x41	0x7fffffff	NUMBER (n) OF STACK UNWINDS AT INSTRUMENTED ROUTINE	pc1-PROGRAM COUNTER OF INSTRUMENTED ROUTINE	pc2-CALLER OF INSTRUMENTED ROUTINE	.....		pcn OF n-1st CALLER OF INSTRUMENTED ROUTINE
0x50	0x10	OBJECT ID	METHOD BLOCK ADDRESS	N/A	N/A		METHOD INVOCATION (JITTED)
0x50	0x10 + 0x80000000	OBJECT ID	METHOD BLOCK ADDRESS	N/A	N/A		METHOD EXIT (JITTED)

FIG. 21A

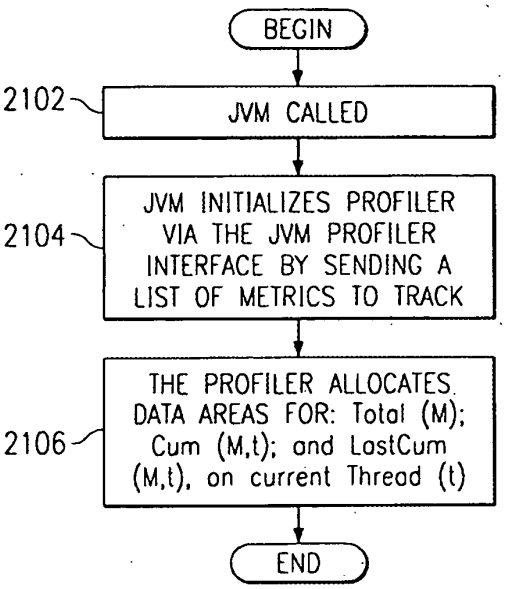


FIG. 21B

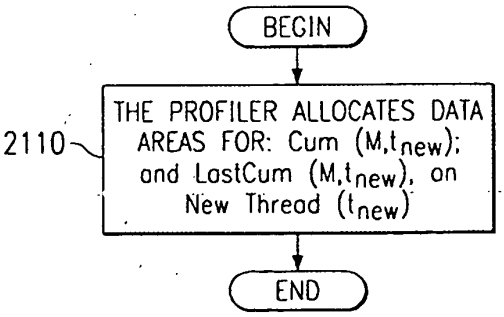


FIG. 22

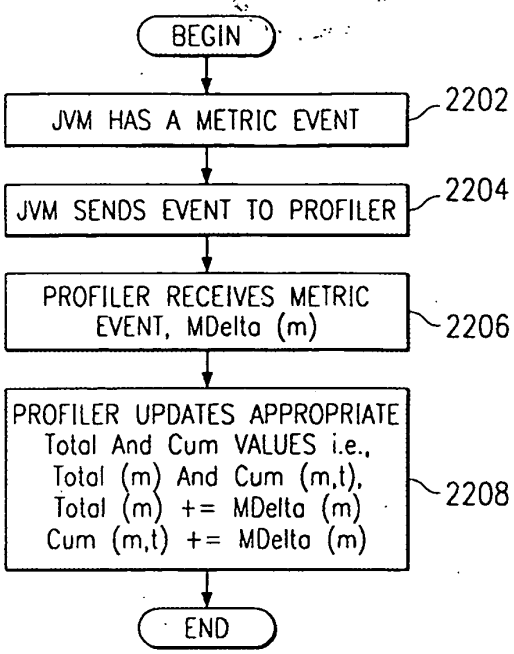


FIG. 23

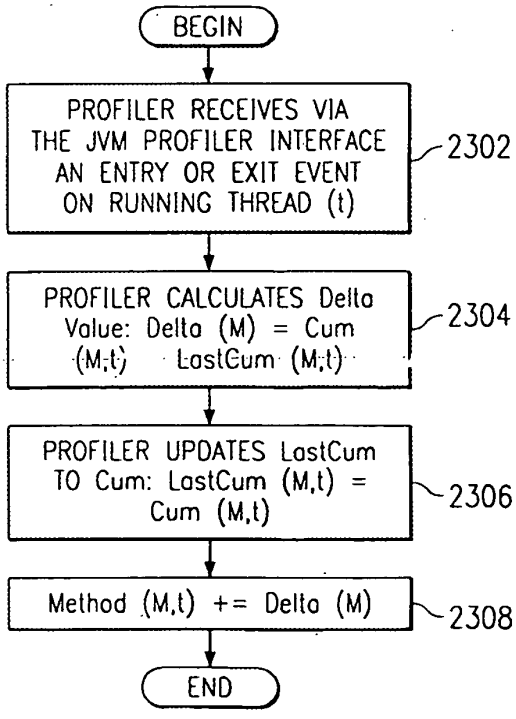


FIG. 24

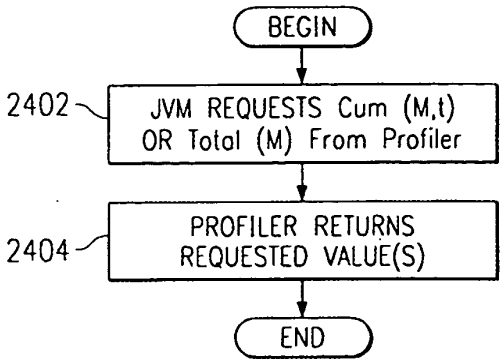

BEGIN
2402
JVM REQUESTS Cum (M,t) OR Total (M) From Profiler
2404
PROFILER RETURNS REQUESTED VALUE(S)
END

FIG. 25

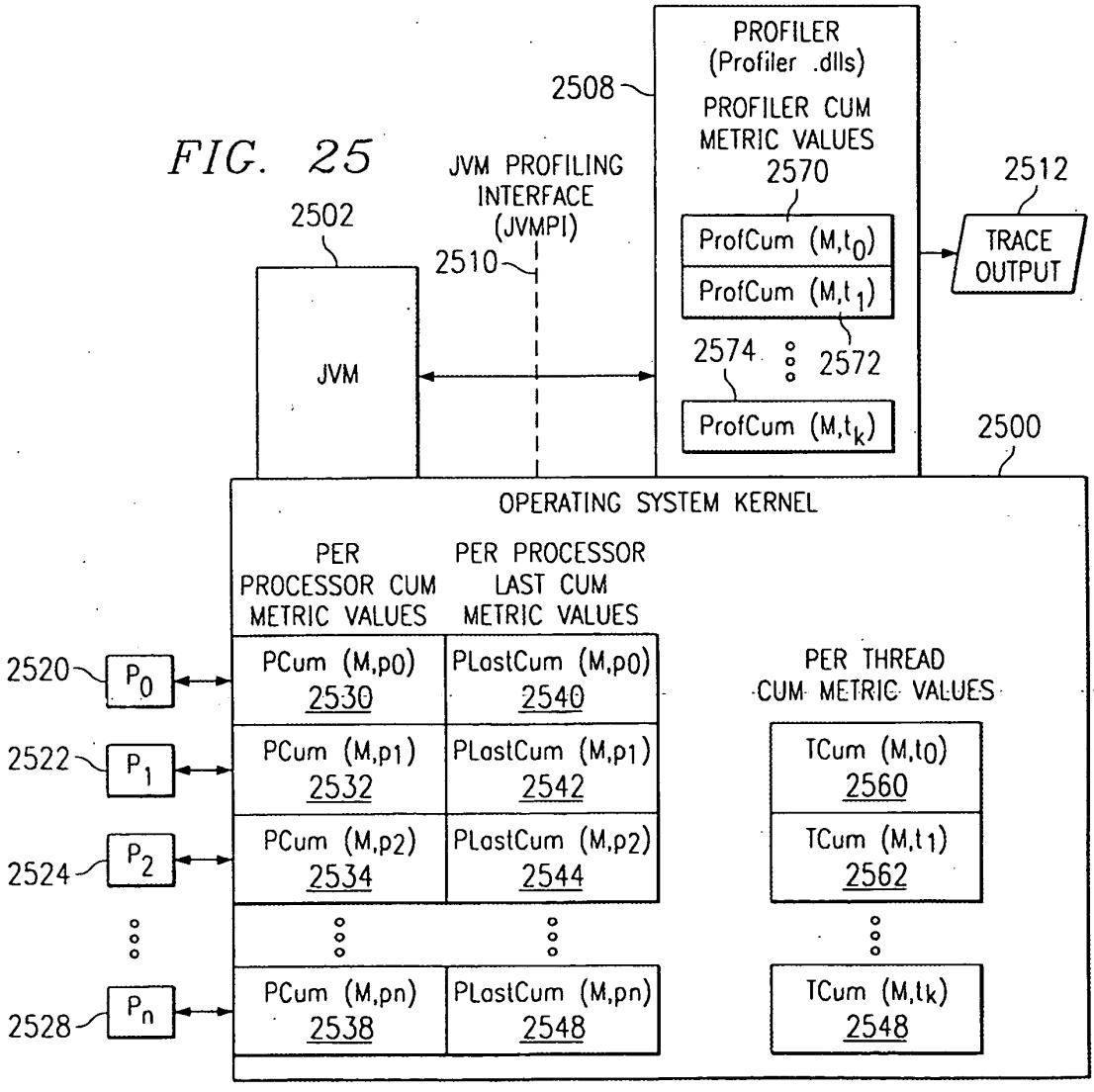

PROFILER (Profiler .dlls)
2508
PROFILER CUM METRIC VALUES
2570
JVM PROFILING INTERFACE (JVMPI)
2502
2510
2512
TRACE OUTPUT
ProfCum (M,t0)
ProfCum (M,t1)
2574
2572
ProfCum (M,tk)
JVM
2500
OPERATING SYSTEM KERNEL
PER PROCESSOR CUM METRIC VALUES
PER PROCESSOR LAST CUM METRIC VALUES
PER THREAD CUM METRIC VALUES
2520
P0
PCum (M,p0) 2530
PLastCum (M,p0) 2540
2522
P1
PCum (M,p1) 2532
PLastCum (M,p1) 2542
TCum (M,t0) 2560
2524
P2
PCum (M,p2) 2534
PLastCum (M,p2) 2544
TCum (M,t1) 2562
2528
Pn
PCum (M,pn) 2538
PLastCum (M,pn) 2548
TCum (M,tk) 2548

FIG. 26A

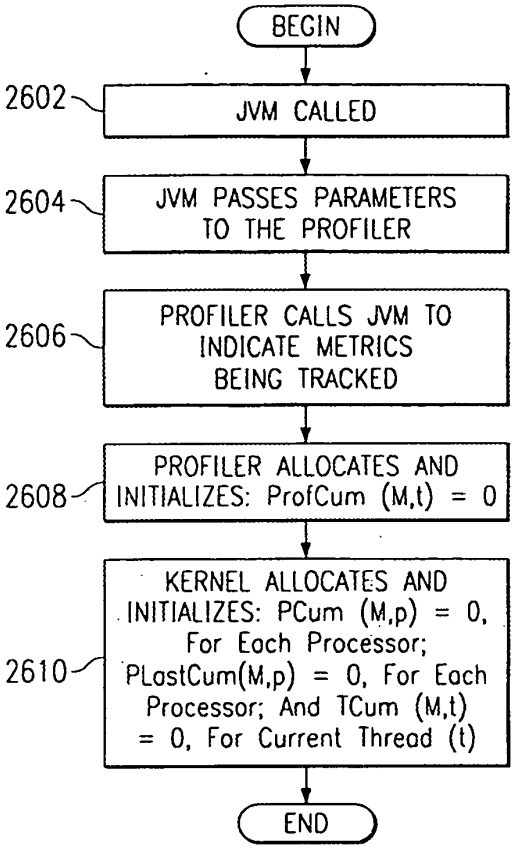


FIG. 26B

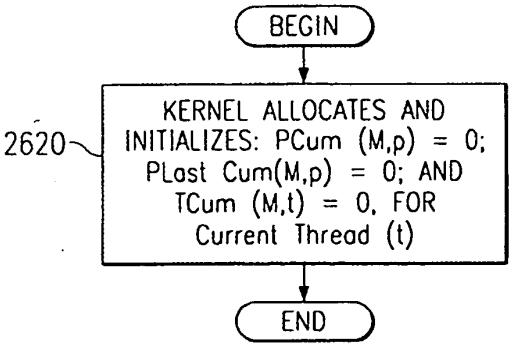


FIG. 26C

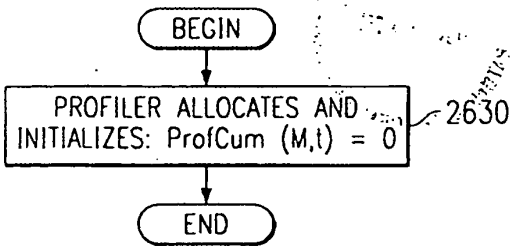


FIG. 27

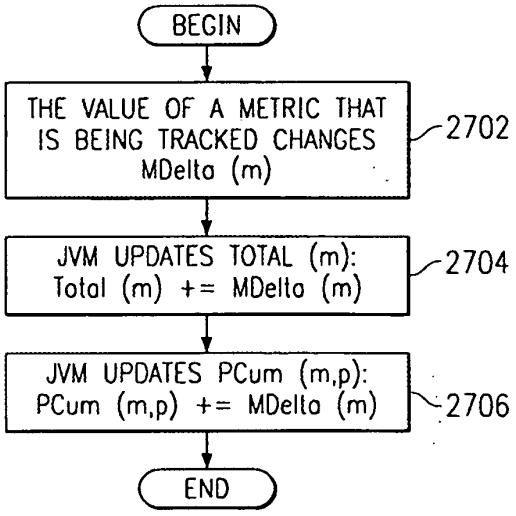


FIG. 28

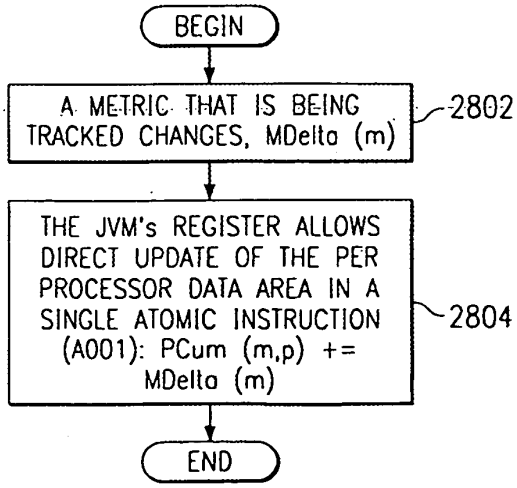


FIG. 29

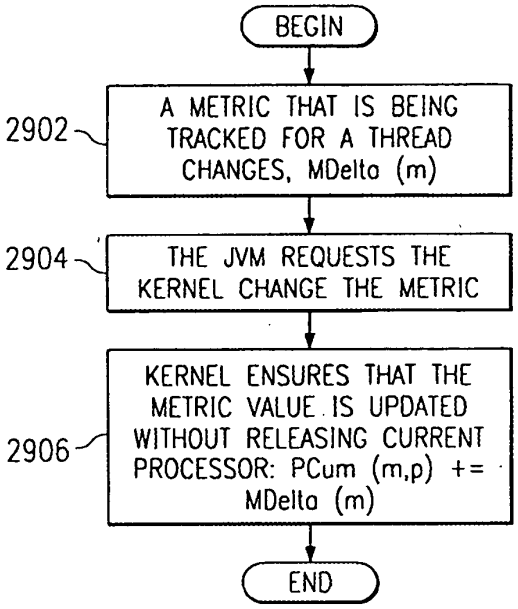


FIG. 30

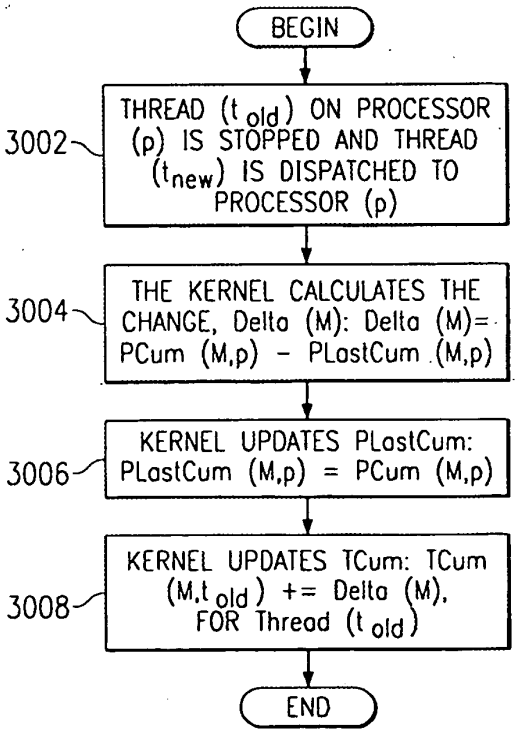
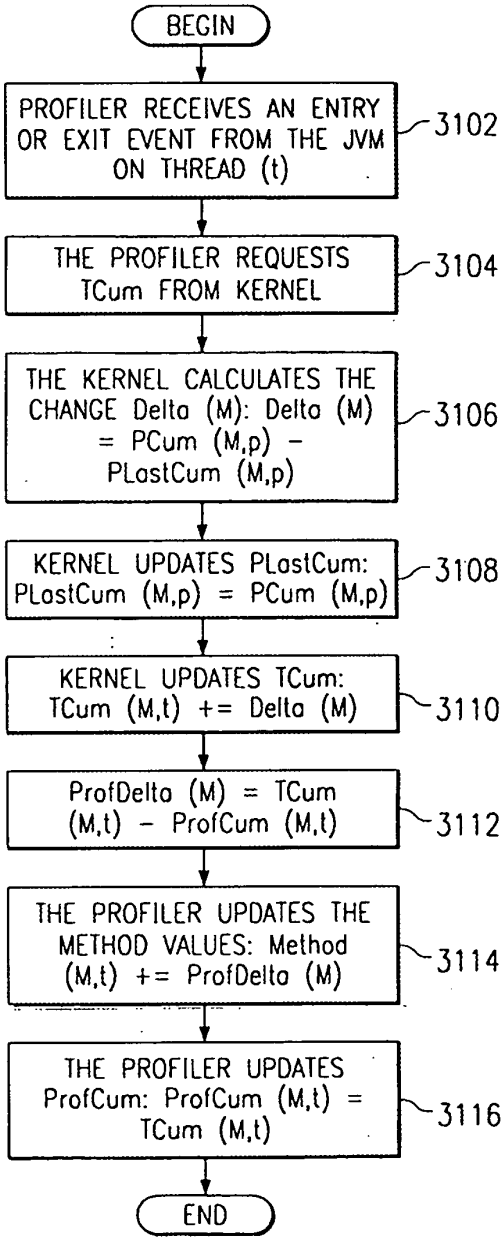



FIG. 31